|  |
| --- |
|  |

**LAB 3: INDEXING STRATEGIES & AGGREGATION FRAMEWORK**

|  |  |
| --- | --- |
| **Track** | Data & AI |
| **Role** | Software Development - Backend |
| **Level** | Level 2 |
| **Course** | NoSQL Databases |
| **Scope** | * Creating and Managing Different Types of Indexes * Understanding Index Performance and Its Impact on Queries * Implemented aggregation pipelines for efficient data analysis. * Optimizing Aggregation Pipeline Stages for Better Efficiency |
| **Prerequisite** | * Familiarity with basic MongoDB commands * Knowledge of data querying and aggregation pipelines * A working MongoDB environment with the provided dataset |

**Objective**

To create and manage indexes and implement aggregation pipelines to optimize query execution, improve database performance, and enhance resource utilization.

**Tasks**

**Dataset (also available in resources): telecom\_dataset\_v2.json**

**Implementing Embedded Documents**

1. **Implementing Referenced Documents**
2. **Creating Different Types of Indexes:**

* Single Field Index
* Compound Index
* Multikey Index
* Text Index
* Geospatial Index
* Unique Index
* TTL (Time to Live) Index

1. **Managing Indexes:**

* List All Indexes
* Drop an Index
* Recreate a Dropped Index
* Check Index Usage
* Drop All Indexes

1. **Understanding Index Performance:**

* Analyze Query Performance Without Index
* Analyze Query Performance with Index
* Evaluate Index Size
* Identify Index Use
* Test Index Impact on Write Operations

1. **Implementing Aggregation Pipelines:**

* Filter by Region and Calculate Average Monthly Bill
* Group by Plan and Count Customers
* Find Maximum Data Usage in Each Region
* Filter Premium Customers and Sort by Data Usage
* Calculate Total Revenue by Region

1. **Optimizing Aggregation Pipeline Queries**

* Move $match Stage Early
* Avoid Unnecessary Fields with $project
* Use Index to Optimize $sort
* Use $limit and $skip Early for Pagination
* Reshape Pipeline to Minimize Computation

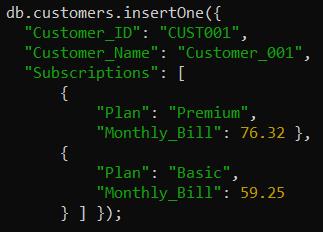
**Outcomes**

* Created and managed different types of indexes to optimize query performance.
* Implemented and optimized aggregation pipelines for efficient data analysis.
* Improved resource utilization by restructuring pipeline stages.
* Measured the impact of indexing on query and write performance.

**Solution**

1. **Implementing Embedded Documents:**

* A customer with multiple subscriptions:
* Use the insertOne operation to add a document with a nested "Subscriptions" array containing subscription details.

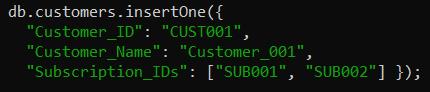


* Output:

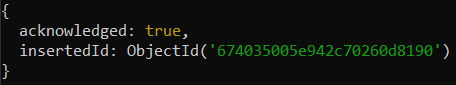


1. **Implementing Referenced Documents:**

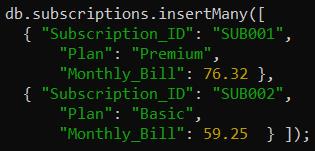
* Use Referenced Documents to link related data across collections using keys.
* Create a customers collection to store customer information with references (Subscription\_IDs) to their subscriptions in another collection.



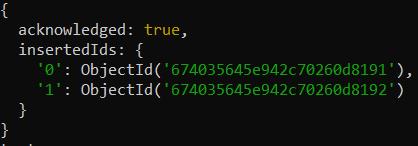
* Output:



* Createa subscriptions collection to store subscription details, each with a unique identifier.



* Output:



1. **Creating Different Types of Indexes:**

* Single Field Index
* Creating an index on the customerId field in ascending order (1).



* Output:



* Compound Index
* Creating a compound index on customerId (ascending) and billingCycle (descending).



* Output:



* Multikey Index
* Creating an index on the tags field, which contains arrays.



* Output:



* Text Index
* Creating a text index on the comments field for full-text search.



* Output:



* Geospatial Index
* Creating a geospatial index on the location field.



* Output:



* Unique Index
* Creating index which ensures no duplicate values for customerId in the telecom collection.



* Output:



* TTL (Time to Live) Index
* Creating a TTL index on the createdAt field.



* Output:

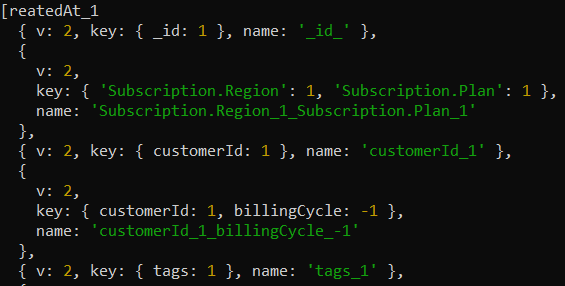


1. **Managing Indexes:**

* List All Indexes
* View all indexes in the telecom collection:



* Output:



* Drop an Index
* Drop the index on Customer\_ID:



* Output:



* Recreate a Dropped Index
* Recreate the dropped index on Customer\_ID:



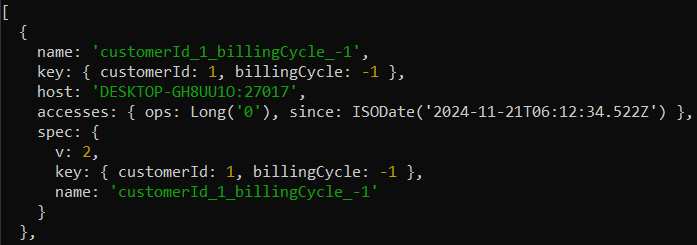
* Output:



* Check Index Usage
* Monitor index usage for optimization:



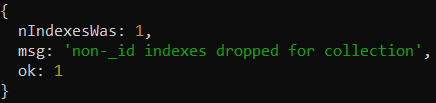
* Output:



* Drop All Indexes
* Drop all indexes except the \_id index:



* Output:

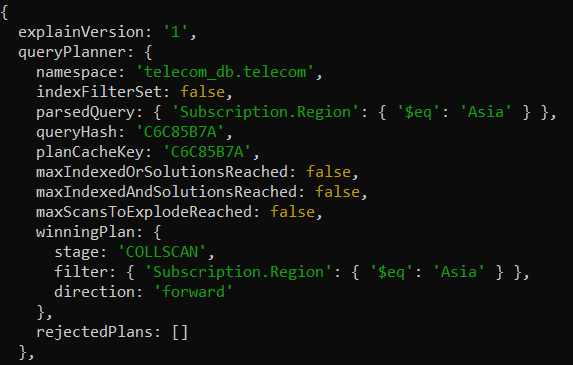


1. **Understanding Index Performance:**

* Analyze Query Performance Without Index
* Run a query filtering by Region and analyze its execution without an index:



* Output:

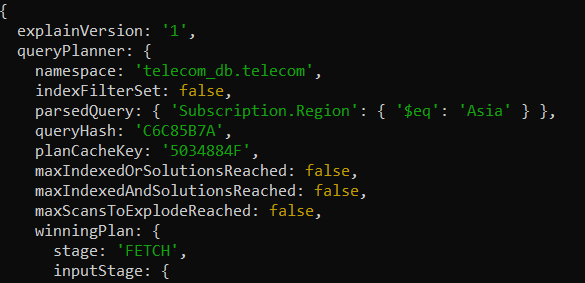


* Analyze Query Performance with Index
* Create an index on region and rerun the query to compare performance:





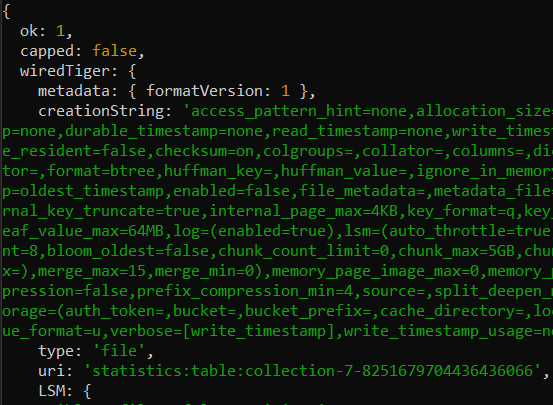
* Output:



* Evaluate Index Size
* Check the size of indexes in the telecom collection:



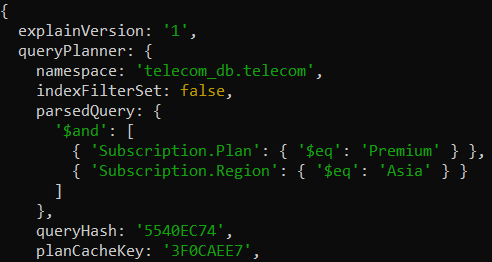
* Output:



* Identify Index Use
* Use the query planner to identify which index is utilized:



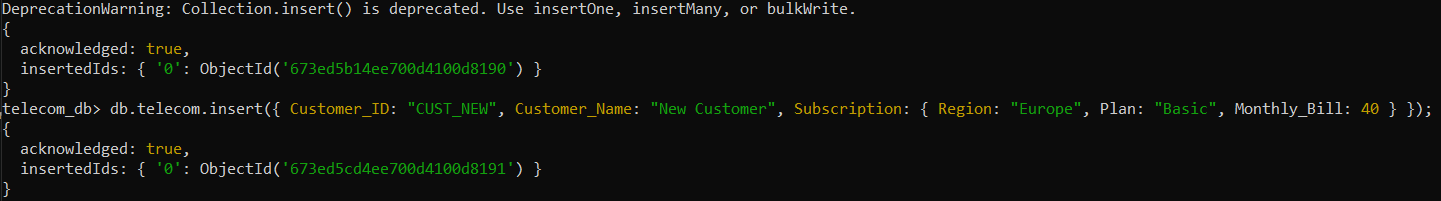
* Output;



* Test Index Impact on Write Operations
* Insert a document and compare performance before and after creating multiple indexes:



* Output;



1. **Implementing Aggregation Pipelines:**

* Filter by Region and Calculate Average Monthly Bill
* Sequentially filter documents by Region and calculate the average Monthly\_Bill:



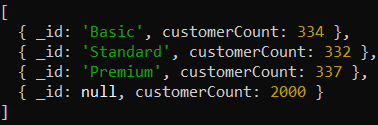
* Output:



* Group by Plan and Count Customers
* Group documents by Plan and count the number of customers in each plan:



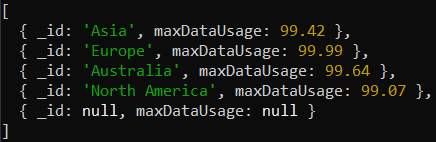
* Output:



* Find Maximum Data Usage in Each Region
* Group by Region and find the maximum Data\_Usage\_GB:



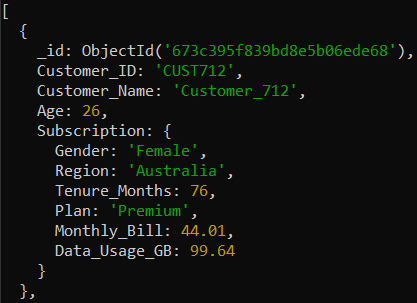
* Output:



* Filter Premium Customers and Sort by Data Usage
* Filter customers on the Premium plan and sort them by Data\_Usage\_GB in descending order:



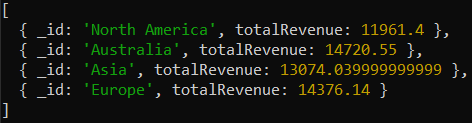
* Output:



* Calculate Total Revenue by Region
* Filter documents for active customers (Tenure\_Months > 12), group by Region, and calculate total revenue (Monthly\_Bill sum):



* Output:

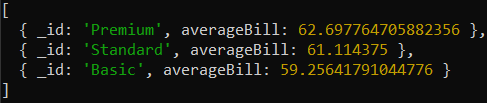


1. **Optimizing Aggregation Pipeline Queries:**

* Move $match Stage Early
* Optimize by filtering documents by Region early in the pipeline:



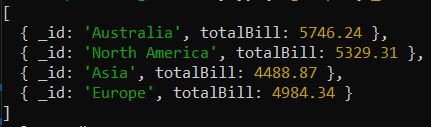
* Output:



* Avoid Unnecessary Fields with $project
* Use $project to limit fields passed to subsequent stages:



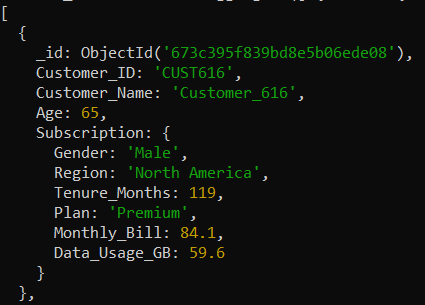
* Output:



* Use Index to Optimize $sort
* Ensure an index exists on Subscription.Tenure\_Months to optimize sorting:



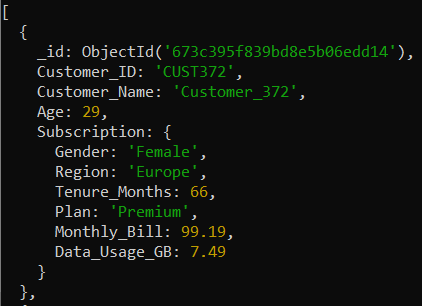
* Output:



* Use $limit and $skip Early for Pagination
* Implement pagination with $limit and $skip early in the pipeline to reduce document volume:



* Output:



* Reshape Pipeline to Minimize Computation
* Replace complex calculations with pre-grouping stages to reduce intermediate document size:



* Output:

